Practical 1: ANN Training and Prediction

Computational Intelligence Assessment: L1 ANN1

Zijin Hong
2020101911
Mathematics and Applied Mathematics
Jinan University - JBJI
`hongzijin@stu2020.jnu.edu.cn`

1 Introduction

This is a report of practical project[1]: Artificial Neural Network Training and Prediction, the catalogue including:

- Task 1's training result and the corresponding analysis
- Task 2's prediction and computation, and required answers

2 Task1: Manual Training an ANN Classifier

In order to solve the problem, we constructed a perceptron consisit of a full-connected layer, using *Sigmoid* function to activate the output as required. The optimizer of the model is SGD (Stochastic Gradient Descent), the loss function is BCE (Binary Cross Entropy) loss for the binary classification task. In order to fit the given data, the input dimension of the linear layer is 3, obtained by considering Study Hrs per Week, Sleep Hrs per Week and Quiz Marks as three variables.

2.1 Results

By setting the learning rate to 0.01, the number of epochs to 3, we train the model with given data and compute the MSE (Mean Square Error) as required.

Epoch	**Loss(1)**	**Loss(2)**	**Loss(3)**	**Loss(4)**	**Loss(5)**
1/3	3.4068	15.4677	6.3916	50.0000	3.0078
2/3	3.2583	14.3871	2.3241	50.0000	5.9446
3/3	3.1757	13.3074	3.9234	50.0000	4.1459
MSE	0.4678	0.4772	0.3331	0.5000	0.4983

Table 1: Five randomly selected training results

To examine the effect of training results of the assigned SGD, we did not set a fixed seed[2] for training. As shown in Table1, we got diverse numerical result. We indicate the model of Loss(1) as our predictor model, the trained model weights are listed in Figure1.

For further analysis, we calculated the average loss and MSE 50 training results. That loss$_{avg} = 23.5656$, MSE$_{avg} = 0.4392$.

[1] The code is available in `https://github.com/Rcrossmeister/CI/tree/main/P1`

[2] In PyTorch, a seed is a parameter that is set to ensure reproducibility of random number generation for experiments or training.

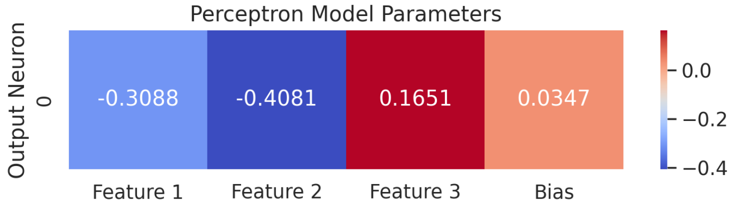


Figure 1: The training weight of perceptron

2.2 Further Analysis

2.2.1 Analysis of Convergence

The model does not seem to be converging smoothly. Ideally, we would expect the loss to decrease consistently over epochs. The fluctuation in loss values indicates that the model might be struggling to find an optimal solution in the given parameter space.

2.2.2 Reasons for Non-Convergence

Model Simplicity. The perceptron is a quite simple model. If the data is not linearly separable, a single-layer perceptron might not be sufficient.

Data Issues. There could be issues with the data such as outliers or it might not be normalized.

2.2.3 Suggestions for Optimization

Adjust Learning Rate. In our training step, the learning rate is empirically set to 0.01. We can try reducing the learning rate. A smaller learning rate might make the model converge more smoothly, albeit slower.

Data Normalization. Normalize the input data to have zero mean and unit variance. This often helps in training neural networks.

Model Complexity. If the data is not linearly separable, consider using a multi-layer perceptron (MLP) with one or more hidden layers.

More Epochs. Train the model for more epochs. Sometimes, the model might need more iterations to converge to an optimal solution.

3 Task2: Churn Rate Prediction and Analysis

For the churn rate prediction, we constructed an artificial neural network (ANN) with two hidden layers, each containing 6 neurons, which is consistent with the number of features selected in the given data preprocessing step. The activation function used for bewteen hidden layers is the *ReLU* (Rectified Linear Unit) function. The output layer contains a single neuron with a *Sigmoid* activation function, suitable for binary classification tasks.

3.1 Results

The model was trained using the Adam optimizer with a learning rate of 0.001, the loss function used for training is the BCE loss, the number of training epochs in 100. The model was evaluated on a test set[3], for further analysis, we set the threshold value of binary classification to 0.5. The results are as follows:

[3] Splited into 20% of the full data

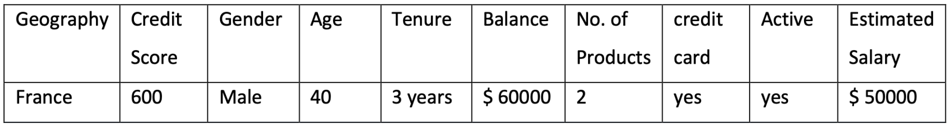
Metric	Value
Confusion Matrix	$\begin{bmatrix} 1431 & 161 \\ 365 & 40 \end{bmatrix}$
Accuracy	73.70%

Table 2: Evaluation metrics for the ANN model on the test set

3.2 Further Analysis and Discussion

3.2.1 Prediction for a New Customer

Given the details of a specific customer:



Geography	Credit Score	Gender	Age	Tenure	Balance	No. of Products	credit card	Active	Estimated Salary
France	600	Male	40	3 years	$ 60000	2	yes	yes	$ 50000

Figure 2: Detail information of the customer

The model's corresponding output is:

Probability that the customer will leave the bank: **0.4808**

Given the predicted probability of 0.4808, which is less than 0.5[4], the model suggests that the customer is more likely to stay with the bank.

3.2.2 Binary Result

If the predicted probability for the customer is greater than or equal to 0.5, the binary result will be 1 (indicating the customer is likely to leave). The detail analysis will be dicussed later.

3.2.3 Confusion Matrix and Accuracy

Confusion Matrix. The confusion matrix indicates that the model predicted 1431 customers would stay, and they did. It also predicted 365 customers would leave, but they didn't. The model did not quite correctly predict any customers who actually left, indicating potential issues with the model's ability to identify the churn.

Accuracy. The accuracy is 73.70%, which means the model has the probability of 73.70% to correctly identify a new customer about their churn rate.

3.2.4 The Normalizaton

Normalization of data is crucial, especially for neural networks. Normalized data ensures that all input features are on a similar scale, which helps the model converge faster and achieve better performance. Without normalization, features with larger scales can dominate the training process, leading to suboptimal results.

4 Conclusion

This report presented the training and evaluation of two neural network models for different tasks. While the perceptron model for Task 1 showed fluctuating training results, the ANN model for Task 2 achieved a high accuracy but showed signs of bias. We analyze the corresponding result with specific question, and obtain the corresponding conclusions.

[4] The default threshold of binary classification task